Optimized trajectory planning for Cybernetic Transportation Systems

Fernando Garrido[1,2], **David González**[1], **Vicente Milanés**[1], **Joshué Pérez**[1], **Fawzi Nashashibi**[1]

[1] *Robotics and Intelligent Transportation Systems (RITS) Team, Inria, 2 Rue Simonne Iff, 75012 Paris, France* {fernando.garrido-carpio, david.gonzalez-bautista, vicente.milanes, joshue.perez_rastelli, fawzi.nashashibi} @inria.fr

[2] *Vedecom institute, 77 Rue des Chantiers, 78000 Versailles, France* fernando.garrido-carpio@vedecom.fr

Abstract:
This paper describes the development of an optimized path planning algorithm for automated vehicles in urban environments. This path planning is developed on the basis of urban environments, where Cybernetic Transportation Systems (CTS) will operate. Our approach is mainly affected by vehicle's kinematics and physical road constraints. Based on this assumptions, computational time for path planning can be significantly reduced by creating an off-line database that already optimized all the potential trajectories in each curve the CTS can carry out. Therefore, this algorithm generates a database of smooth and continuous curves considering a big set of different intersection scenarios, taking into account the constraints of the infrastructure and the physical limitations of the vehicle. According to the real scenario, the local planner selects from the database the appropriate curves from searching for the ones that fit with the intersections defined on it. The path planning algorithm has been tested in simulation using the previous control architecture. The results obtained show path generation improvements in terms of smoothness and continuity. Finally, the proposed algorithm was compared with previous path planning algorithms for its assessment.

Keywords: Path planning, database, intelligent algorithm, cost function, optimization

1. INTRODUCTION

Nowadays, road transport remains as the main way of moving both people and goods in the world. In 2012, road transport accounted for 44.9% of the total goods transport and 82.4% of the passenger transport among cars, powered two-wheelers and buses in the European Union (EU) (European Commission, 2014). Having this in mind, the research on this domain in order to improve the efficiency and safety for road vehicles constitutes an open challenge to be further investigated.

Specifically talking about people transport, almost 25,700 road fatalities were reported in the EU during 2014. It means a decrease of 1% compared to 2013 but 18% fewer than in 2010, indicating that the decrease rate on road facilities has been dramatically reduced (European Commission, 2015). This points out to a real problem affecting modern society that of course has to be improved.

Intelligent Transportation Systems (ITS) rise as a research field to improve transportation from the points of view of traffic flow and safety (Shladover et al., 2011). It can be defined as the different intelligent systems that can be equipped over a road transportation in order to improve its ability toward a safer and smarter way of moving both people and goods. One of the final goals of the ITS domain in the road transport system is to be able to deploy fully automated vehicles (Parent, 2013).

In the last decades, automated vehicles have turned into one of the most promising research lines because of their great potential to improve the transportation system (Xu et al., 2012). There is still a long way to go before having fully automated vehicles driving on public roads. However, significant results have been achieved, since the pioneers first miles driven autonomously by a Mercedes van in the 90s (Dickmanns et al., 1994) up to the most recent years experiments. Among them, Google driverless cars demonstrations, VisLab Intercontinental Autonomous Trip (Broggi et al., 2010) , Karlsruher Institut für Technologie (KIT) automated Mercedes car first trip emulation (Ziegler et al., 2014) or, in the urban context, the results achieved by the INRIA (Institut National de Recherche en Informatique et Automatique) RITS (Robotics and Intelligent Transportation Systems) team based on the door to door novel concept: cybercars, also called CTS (Milanés et al., 2014).

Among the different scenarios where automated vehicles have to operate, urban areas represent the most complex environments because of the different road actors (pedestrians, cyclists, etc) that can be found when driving, especially in unexpected situations. For uncontrolled environments, trajectory planning plays a key role for

determining the best path the automated vehicle should follow considering both: vehicle and road constraints.

In this paper, a new path planning strategy is proposed to achieve in real time a safe, smooth and continuous trajectory generation to be tracked by low-speed vehicles in urban scenarios. As the computational cost for the real-time path planning of the potential curves that the vehicle can carry out could be too high, an off-line evaluation has been applied. Assuming low-speed urban environments as the ones where cybercars will operate, vehicle behavior is not affected significantly by vehicle dynamics and the planning is mainly affected by vehicle's kinematics and physical road constraints. In this stage a database of different curves is generated in order to consider a big enough set of possible intersection scenarios, considering a reasonable and safe enough maximum distance. It will take into account physical limitations of the vehicles, the characteristics associated to the infrastructure and based on a cost-function criterion it will choose the smoothest and more comfortable curves. Then, during the real-time planning stage, the configuration of the best feasible curves are loaded from the database generating the smooth and continuous paths according to the different intersections that compose the real scenario.

The rest of the paper is structured as follows. Section II describes the path planning problems for automated vehicles. The path planning algorithm combining both off-line path generation and local planning evaluation is presented in section III. Simulation results are included in Section IV. Finally, some concluding remarks and future work are given.

2. PROBLEM DESCRIPTION

The motion planning problem can be defined as the search for a feasible and collision-free trajectory that allows a vehicle to go from a point A to a point B (Latombe, 1999). During the last three decades, path planning has been widely applied in two domains: industrial robotics and automated systems (Strandberg, 2004). Regarding the automated vehicles field, mobile robots are increasingly being employed in many automated environments (Raja and Pugazhenthi, 2012), such as service robots, guidance vehicles and for exploration reasons. In addition to mobile robots, automated vehicles represent another well-known application area of motion planning. In this period, both research centres and industry have been working in the development of motion planning techniques for autonomous driving, trying to improve transport efficiency and safety (Gu and Dolan, 2012).

There exist some clear differences between path planning for autonomous vehicles and path planning for robots. These differences basically concern to the vehicle, the infrastructure and its limitations. First of all, a vehicle presents nonholonomic constraints related to its kinematic that affect the planning stage. Unlike robots, automated cars such as the cybercars don't present three degrees of freedom, so they cannot move freely in all the three degrees of motion because of the steering constraints (Pasha, 2003). Furthermore, there are some constraints in the environment: trajectories have to lie within road limits, taking into account the surrounding road actors (pedestrians, cyclists, etc) and respecting the driving laws.

Based on the availability of information about the environment, path planning algorithms can be classified into off-line and on-line strategies (Raja and Pugazhenthi, 2012). Off-line algorithms are used when there is a previous knowledge of the surrounding environment, whereas on-line algorithms are implemented to react in real-time to unforeseen changes in the environment.

These components compose the decision stage of the control architecture for the INRIA RITS team Cybernetic Transportation Systems (CTS), detailed in (González and Pérez, 2013; Pérez Rastelli et al., 2014; González et al., 2014). So far, global planner gave inaccurate information of the environment. In our approach, an accurate global path is generated by taking into account the current vehicle position and its final destination. This module is capable to create a virtual railroad which will be followed by the Cybercar. Meanwhile, the local planner ensures a real time, safe and comfortable trajectory throughout the path, smoothing the raw path by considering the kinematics of the vehicle and the road constraints.

There exist several path generation techniques that can be used in the planning stage. The A* algorithm and state lattices are graph search based algorithms used in path finding; Rapidly-exploring Random Tree (RRT) and Enhanced RRT (RRT*) are sampling-based algorithms applied in planning with real-time constraints; Clothoids, polynomial curves, splines and Bézier curves are interpolating algorithms that smooth a previous path composed of a set of way-points.

Among them, we use **Bézier curves** because of their simplicity and low computational load which make it more suitable than the other path generation techniques (Han et al., 2010), allowing us a fast trajectory computation. This is a needed feature that has to be fulfilled by the proposed off-line planner, because it analyzes a great set of feasible curves in order to select the optimal one. In addition, it also constitutes a requirement for the local planner in order to generate the selected trajectory from the database and send it in real time to the control stage. Then, the goal of the planning system is to build a continuous, smooth and safe path that will be tracked by the automated vehicle.

2.1 Bézier curves definition and properties

Bézier curves are polynomial curves that use information of checkpoints, called control points, to be generated. The basis of these curves are the Bernstein polynomials. Equation 1 presents the mathematical definition for a n-degree Bézier curve generation.

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i P_i \quad t \in [0,1] \tag{1}$$

where n is the degree of the polynomial equation and P_i are the control points that define the curve.

In (Han et al., 2010) some features of the Bézier curves are described. The most relevant for the work done are explained below.

- Bézier curves always start on the first control point P_0 and they end at the last control point P_n.
- The direction vector that defines the beginning of the curve is described by $\overrightarrow{P_0P_1}$ and the direction vector that defines the end of the curve is described by $\overrightarrow{P_{n-1}P_n}$.
- The curves are fully symmetric, i.e., if the control points are reversed, we get the same curve.
- The curves will be always framed within the convex hull defined by the control points, i.e., the one formed by the outermost control points.
- The behaviour of the curve concavity is consistent with the concavity formed by the control points.

Thus, we will test our trajectory generation approach in urban scenarios where different kind of curves can be found. The results obtained will be compared with the previous works developed in the INRIA RITS team for intersection scenarios (Pérez Rastelli et al., 2014).

3. PATH PLANNING ALGORITHM

This work proposes a two-stage local planning system. In the first stage, a database of different curves is generated in order to consider all the possible intersection scenarios. Limitations of the ego-vehicule and the characteristics associated to the infrastructure were taken into account in the database generation. The smoothest and more comfortable curves are chosen based on a cost-function criterion. In the second stage, the configuration of the best feasible curve for each intersection is loaded to generate the paths according to the different intersections that compose the real scenario.

3.1 Off-line planning stage

The best trajectory for a given curve can be off-line generated with the proposed intelligent algorithm. Its goal is to search the best possible trajectory for each kind of intersection of a given scenario.

Let's consider a generic curve as the one depicted in Figure 1. It shows an intersection scenario described geometrically with the angle α of the intersection, the points G_{n-1}, G_n and G_{n+1} provided by the global planner and the road width R_W defining the road lane centre. In this study, only third and fourth degree bezier curves are considered. Modifying the control points along the road before and after the bend, the most suitable path can be found. For doing so, we define segments d_0 and d_4 that represent the distance from the intersection midpoint to the first (P_0) and last (P_n) control point, i.e., to the first and last curve points respectively. Meanwhile, the segments d_1 and d_3 represent the distance from the midpoint to the internal control points. These control points can be moved laterally through the road centre, being d_{lat} the lateral displacement for the internal control points (P_1 and P_{n-1}) and, in the case of having a 4th degree Bézier curve, d_2 represents the lateral displacement for the middle control point (P_2). Finally, the points L_0, L_1, L_2 represent the internal and R_0, R_1, R_2 the external road constraints. These points define the Convex Hull where the control points have to be placed in order to ensure that the vehicle remains into the lane.

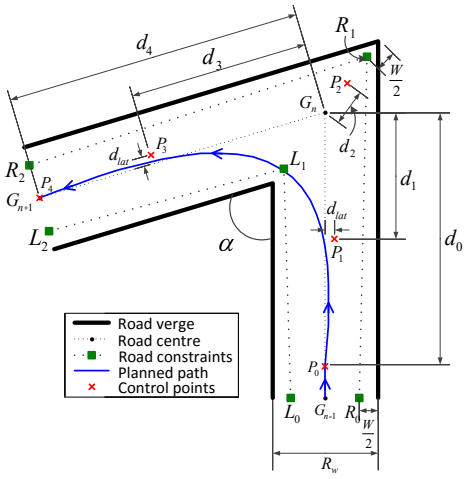

Fig. 1. Curve representation on an intersection scenario

The intelligent algorithm developed in the off-line planning stage is introduced in Algorithm 1.

Algorithm 1 Paths database generation algorithm

1: **for** each intersection (changing α, d_0, d_4) **do**
2: **Calculate** road limits and road constraints
3: **Changing** d_1, d_3, d_{lat} and d_2
4: **Calculate** control points location
5: **Check** if the curve is feasible by the vehicle
6: **Check** if the continuity constraints are fulfilled
7: **while** the curve is not totally generated **do**
8: **Generate** curve points
9: **Calculate** nearest point to constraint L_1
10: **Check** if the car doesn't invade the sidewalk
11: **Calculate** curve's fitness with cost function
12: **Check** if this curve improves the best one
13: **Save** the best curve's configuration in the database

As the goal is to generate a complete enough database of safe and continuous feasible curves for a big set of intersection scenarios, in line 1 of Algorithm 1 the parameters that define the intersection are settled: the intersection angle is changed from a minimum value of 5 degrees (very closed curve) to a maximum value of 180 degrees (straight line) every 5 degrees. Meanwhile, the distance from the intersection midpoint G_n to the external points G_{n-1} and G_{n+1}, i.e., the length of both segments, is changed from a minimum distance, that is 4 meters, to a far enough distance of 40 meters. The length of both segments is changed every 0.2 meters, what is acceptable due to the positioning systems maximum error is considered to be around 0.3 meters (Hasberg et al., 2012).

Once the parameters that define the intersection are defined, the road limits are calculated according to the road width (line 2, Algorithm 1). In addition, road constraints will be generated, considering an internal separation of half the car width with respect to the road limits. In this sense, the first (P_0) and last (P_n) control points are located over the straight lines before and after the intersection midpoint (based on the 1st property of Bézier curves) at d_0 and d_4 meters, whereas the control points P_1 to P_{n-1} are set in such a way that the convex hull does not surpass the drivable area of the vehicle. Meanwhile, the longitudinal distances, d_1 and d_3, are changed from

the midpoint of the intersection G_n to the location of the external control points (d_0 and d_4) (lines 3, 4). In addition, the internal control points are also moved laterally from the road centre to the bounds of the convex hull. These lateral displacements of the control points are represented by d_{lat} and d_2. The location of the control points are described by Equation 2 with positive sign if it is a left turn, or negative if it is a right turn.

$$P_i = G_n + d_1 \frac{G_{ext} - G_n}{\|G_{ext} - G_n\|} \pm latDisp \cdot \cos\left(\alpha - \frac{\pi}{2}\right) \quad (2)$$

where G_{ext} is the external intersection point, that is, either G_{n-1} or G_{n+1}, α is the angle between the two segments that define the intersection and $latDisp$ is the lateral displacement, that will be either d_2 for the middle control point of the curve (P_2) or d_{lat} in any other case.

After that, it is necessary to check if the curve is feasible by the vehicle (line 5, Algorithm 1), that is, the curvature in all Bézier curve must be smaller or equal to the maximum curvature feasible by the vehicle. The maximum curvature feasible is when the angle of the wheel is maximum using the Ackerman model. The analytical method is used to calculate the curvature k at each point t of the parametric Bezier $\overrightarrow{B}(t)$. Considering two-dimensional curves, the equation to obtain the curvature is defined as follows:

$$k(t) = \frac{\overrightarrow{B}'(t) \times \overrightarrow{B}''(t)}{\left\|\overrightarrow{B}'(t)\right\|^3} \quad (3)$$

In order to get a smooth trajectory, a constraint in the curvature at the beginning and at the end of the curve is defined. We consider that the curvature at these points should be as close to zero as possible.

Equation 1 is used to generate the n-degree Bézier curves, where n is the degree of the polynomial, until the curve is totally generated according to the number of divisions defined (line 8, Algorithm 1). Furthermore, the curve's point nearest to the road constraints point L_1 is calculated. Then, we verify that the car doesn't invade the sidewalk (line 10, Algorithm 1). It must be ensured a minimum distance of half of the vehicle width ($W/2$) between the joint of the segments and the nearest point of the curve (represented by the point L_1).

The cost function will take into account all the possible parameters independently of vehicle dynamics (line 11, Algorithm 1). All the generated curves are evaluated, selecting the optimal one according to our cost function (Line 12, Algorithm 1).

Previous works in the INRIA RITS team (Pérez Rastelli et al., 2014) used a cost function in the intelligent algorithm that took into account the measure of curvature k in the three most critical points: at the beginning, in the middle and at the end of the curve. In our approach, we take into account the curvature at every point of the curve in order to have a better estimation of the optimality of the curve. Besides using the curvature measurement k, the curvature derivative measurement k' is also taken into account at every point of the curve. This evaluation allows to minimize sudden changes on the curvature, leading to a smoother and more comfortable path. Equation 4 defined in (Walton et al., 2003) is used to calculate the derivative of the analytical curvature at each point of the curve, where the parametric curve is defined by the set of points $\overrightarrow{B}(t) = (x(t), y(t))$ for a real $t \in [0, 1]$

$$k'(t) = \frac{q(t)}{\left\|\overrightarrow{B}'(t)\right\|^5} \quad (4)$$

where

$$\begin{aligned} q(t) = &\left\{\overrightarrow{B}'(t) \cdot \overrightarrow{B}'(t)\right\}\left\{\overrightarrow{B}'(t) \times \overrightarrow{B}'''(t)\right\} \\ &-3\left\{\overrightarrow{B}'(t) \times \overrightarrow{B}''(t)\right\}\left\{\overrightarrow{B}'(t) \cdot \overrightarrow{B}'(t)\right\} \end{aligned} \quad (5)$$

Hence, the goal of this cost function is to minimize both curvature and its derivative in all the points defining a curve, penalizing abrupt changes of the curvature and trying to find the minimum possible curvature to have a smooth and safe path. Then, the curve's fitness is calculated with Equation 6, and the best curve for an intersection will be the one that minimizes this fitness value.

$$fitness = \sum_{i=0}^{m} w_{ki} |k_i| + w_{k'i} |k'_i| \quad (6)$$

where m represents the number of points defining the curve, w_{ki} is the weight assigned to the curvature and $w_{k'i}$ the one assigned to the curvature derivative.

Several weights of the curvature and curvature derivative have been taken into account in the definition of the cost function, trying to give more importance either to the curvature or to the curvature derivative. The results obtained with the several cost functions reflected that the best weighting was $w_{ki} = w_{k'i} = 1$.

This information is saved into the database in order to be loaded later in the real-time local planner to generate the corresponding curve for each intersection (line 13, Algorithm 1).

3.2 Real-time planning

The goal of the local planner is to select the proper curve from the database according to the specific urban intersections scenario.

The local planner receives the points defining the urban scenario from the global planner (points G_{n-1}, G_n and G_{n+1} in Figure 1). At each execution step, the local planner will read the global planner's following point to know if it is the end of the path or an intersection. In the last case, it forms the segments defining the intersection scenario reading the current and the two following points of the global planner. In addition, the angle between them is calculated.

Since the database cannot consider all the possible angles of a real intersection scenario, a fitting is performed to load the nearest closed curve from the database. For instance, if a 91.7 degrees intersections is read from the global planner, as the database path's rate of change is 5 degrees, the optimal curve for a 90 degrees configuration will be read. A similar fitting is performed with the length of both segments, considering the shorter length. This length is

changed every 2 meters. Thus, searching the database for the most similar curve to the real scenario in hand.

4. EXPERIMENTAL RESULTS

This section presents the simulation experiments to validate the improvement of the proposed planning system in comparison with the previous work. To that end, the planning module has been developed in RTMaps [1] , and the generated component has been tested with Pro-Sivic simulator [2] . This software simulates complex scenarios such as urban environments, allowing us to use the perception information and the models of the vehicles.

The experiments have been carried out on several urban intersection scenarios. Figure 2 shows in the upper part both the planned and tracked paths with the previous team's implementation and the proposed method. Whereas, in the lower part the curvature and its derivative are presented. The urban intersections tested are from the left to the right side of the figure: 150°, 120°, 90°, 60° because these are the most representative kind of intersections that a vehicle can find while driving.

In the path figures the black solid line shows the (Pérez Rastelli et al., 2014)'s path, while the gray solid one shows the path generated with the proposed algorithm. The dotted lines represent the paths tracked by the vehicle. All these paths have been generated with 4th degree Bézier curves. It must be noted that with 3rd degree Bézier curves there are few curves that fulfill the algorithm continuity requirement, then this constraint has to be relaxed and it might not be assured to have continuity in the joint between two intersections. Thus, 4th degree Bézier curves are more convenient. Meanwhile, in the lower figures, the black and gray solid lines represent the curvatures for both the (Pérez Rastelli et al., 2014) and the proposed planning methods, whereas the dashed ones represent the curvature derivative respectively.

Table 1 summarizes the most relevant information of the prior experiments. A reduction on the execution time is reached. Considering that for each intersection scenario the previous algorithm's has around $2,000$ iterations and ours has around $10,400,000$ iterations, i.e. $5,000$ times more, the execution time has been reduced a 34% in the worst scenario. It can also be noticed an improvement on the curvature in all the scenarios. Regarding to the maximum curvature, it improves from 32% for a 60° intersection up to a 62% for a 150° intersection. It can also be shown a reduction on the mean curvature difference. Furthermore, the derivative of the curvature presents the same behaviour. In Figure 2 we can appreciate that the curvature and its derivative with the presented method (gray lines) have lower values than the obtained with the other method (black lines), improving the continuity and leading us to a smoother and more comfortable path planning approach.

5. CONCLUSIONS AND FUTURE WORKS

In this work, a new path planning algorithm has been presented for CTS in urban scenarios. As these vehicles operate at low-speed, an optimization in the path generation strategy is developed without considering vehicle dynamics, that is, considering the kinematics of the vehicle and the road constraints.

The proposed approach improves the path in terms of continuity, providing a smoother and more comfortable trajectory tracking. In addition, the proposed algorithm requires less computation cost to achieve the improvement in the trajectory generation.

As future work, the proposed algorithm will be tested on consecutive curves in urban scenarios, dealing with the continuity problem in the joint of the curves when there is a limited space between them. A novel algorithm to consider several intersections will be introduced to optimize the trajectory.

After that, vehicle dynamics will be considered in the local planner to tune the trajectories charged from the database generated during the off-line planning. Some of its parameters will be the speed, steering angle, friction forces, among others.

In addition, this algorithm has to be developed in other urban scenarios such roundabouts. Furthermore, we will consider other path generation strategies, such as splines or clothoids, in order to make a comparison analysis of the impact of the different path generation methods. In order to improve the trajectory tracking, some works in the controller stage will be considered. Finally, the experiments showed will also be reproduced with real platforms.

[1] www.intempora.com
[2] www.civitec.com

ACKNOWLEDGMENTS

Authors wants to thank the VEDECOM institute and the EU CityMobil-2 project for its support in the development of this work.

REFERENCES

Broggi, A., Medici, P., Cardarelli, E., Cerri, P., and Giacomazzo, A. (2010). Development of the control system for the vislab intercontinental autonomous challange. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference*.

Dickmanns, E., Behringer, R., Dickmanns, D., and Schiehlen, J. (1994). The seeing passenger car vamors-p. In *1.*

European Commission (2014). *EU transport in figures. Statistical pocketbook 2014.*

European Commission (2015). *Road safety in the European Union. Trends, statistics and main challanges.*

González, D. and Pérez, J. (2013). Control architecture for cybernetic transportation systems in urban environments. In *IEEE Intelligent Vehicles Symposium (IV)*.

González, D., Pérez Rastelli, J., Lattarulo, R., Milanés, V., and Nashashibi, F. (2014). Continuous curvature planning with obstacle avoidance capabilities in urban scenarios. In *IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*.

Gu, T. and Dolan, J.M. (2012). On-road motion planning for autonomous vehicles. In *Intelligent Robotics and Applications. 5th International Conference, ICIRA*.

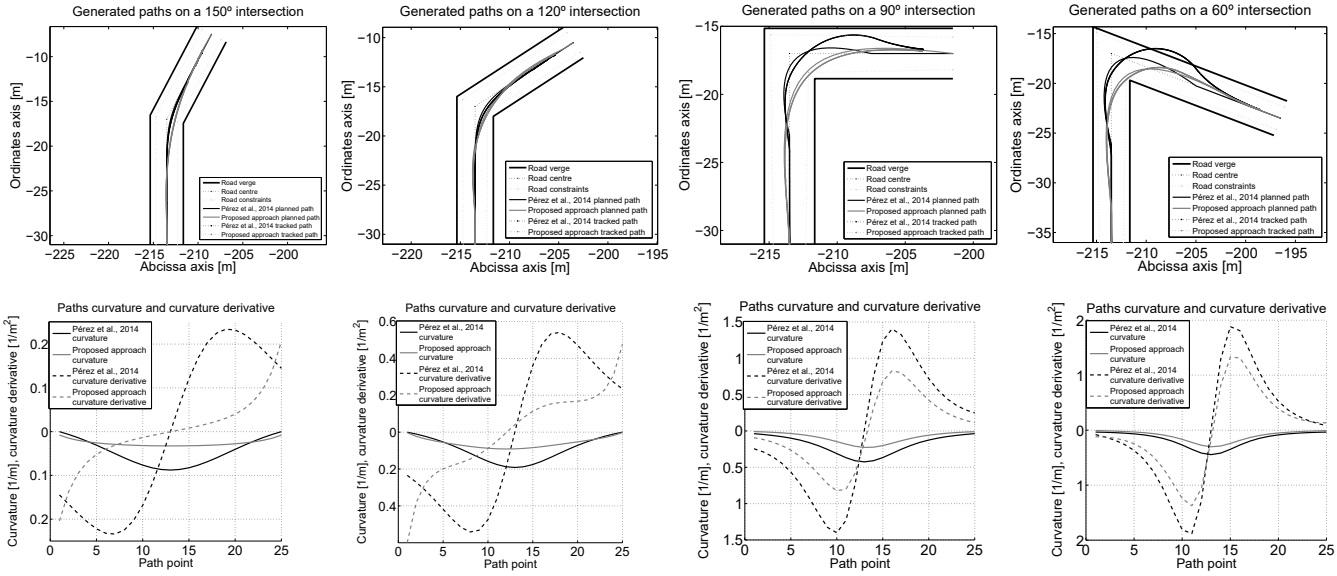

Fig. 2. Path planning experiments over intersection urban scenarios

Table 1. Experiments summary

Intersection scenario	**Algorithm**	**Measurements**				
		time (ms)	$\|\mu_k\|$ $(1/m)$	$\|k_{max}\|$ $(1/m)$	$\|\mu'_k\|$ $(1/m^2)$	$\|k'_{max}\|$ $(1/m^2)$
150°	Pérez et al., 2014	48	0.0466	0.0878	0.1738	0.2340
	Proposed approach	13	0.0259	0.0327	0.0560	0.2061
120°	Pérez et al., 2014	24	0.0925	0.1917	0.3759	0.5415
	Proposed approach	13	0.0583	0.0915	0.1936	0.5997
90°	Pérez et al., 2014	18	0.1893	0.4256	0.7504	1.4033
	Proposed approach	12	0.0909	0.2267	0.4247	0.8275
60°	Pérez et al., 2014	22	0.1637	0.4453	0.7815	1.8833
	Proposed approach	13	0.1020	0.3021	0.5709	1.3745

Han, L., Yashiro, H., Nejad, H.T.N., Do, Q.H., and Mita, S. (2010). Bzier curve based path planning for autonomous vehicle in urban environment. In *IEEE Intelligent Vehicles Symposium*.

Hasberg, C., Hensel, S., and Stiller, C. (2012). Simultaneous localization and mapping for path-constrained motion. *IEEE Transactions on Intelligent Transportation Systems*, 13(2).

Latombe, J.C. (1999). Motion planning: A journey of robots, molecules, digital actors, and other artifacts. *The International Journal of Robotics Research*.

Milanés, V., Marouf, M., Pérez, J., González, D., and Nashashibi, F. (2014). Low-speed cooperative car-following fuzzy controller for cybernetic transport systems. In *IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*.

Parent, M. (2013). Automated vehicles: autonomous or connected? In *IEEE 14th International Conference on Mobile Data Management*.

Pasha, A. (2003). *Path planning for nonholonomic vehicles and its application to radiation environments*. Master's thesis, University of Florida.

Pérez Rastelli, J., Lattarulo, R., and Nashashibi, F. (2014). Dynamic trajectory generation using continuous-curvature algorithms for door to door assistance vehicles. In *IEEE Intelligent Vehicles Symposium (IV)*.

Raja, P. and Pugazhenthi, S. (2012). Optimal path planning of mobile robots: A review. *International Journal of Physical Sciences*, 7(9), 1314 – 1320.

Shladover, S.E., Barth, M.J., and Zhang, W.B. (2011). Engaging the international community: Research on intelligent transportation systems (its) applications to improve environmental performance. Technical report, University of California. California PATH Program, U.C. Berkeley. Center for Environmental Research and Technology, U.C. Riverside.

Strandberg, M. (2004). *Robot Path Planning: An Object-Oriented Approach*. Ph.D. thesis, Automatic Control Department of Signals, Sensors and Systems Royal Institute of Technology (KTH) Stockholm, Sweden.

Walton, D., Meek, D., and Ali, J. (2003). Planar G^2 transition curves composed of cubic Bézier spiral segments. *Journal of Computational and Applied Mathematics*.

Xu, W., Wei, J., Dolan, J.M., Zhao, H., and Zha, H. (2012). A real-time motion planner with trajectory optimization for autonomous vehicles. In *IEEE International Conference on Robotics and Automation*.

Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., Dang, T., Franke, U., Appenrodt, N., Keller, C.G., Zaus, E., Herrtwich, R.G., Rabe, C., Pfeiffer, D., Lindner, F., Stein, F., Erbs, F., Enzweiller, M., Knppel, C., Hipp, J., Haueis, M., Trepte, M., Brenk, C., Tamke, A., Ghannat, M., Braun, M., Joss, A., Fritz, H., Mock, H., Hein, M., and Zeeb, E. (2014). Making bertha drive - an autonomous journey on a historic route. *IEEE Intelligent Transportation Systems Magazine*.